

# A KZ Reduction Algorithm

Jinming Wen and Xiao-Wen Chang

**Abstract**—The Korkine-Zolotareff (KZ) reduction is one of the often used reduction strategies for decoding lattices. A KZ reduction algorithm involves solving shortest vector problems (SVPs) and basis expansion. In this paper, first we improve the commonly used Schnorr-Euchner search strategy for solving SVPs. Then, we derive upper bounds on the magnitudes of the entries of any solution of a SVP when its basis matrix is LLL reduced. These upper bounds only involve the parameter of the LLL reduction and the dimension of the solution and they are useful for analyzing the complexity of the basis expansion in a KZ reduction algorithm. Finally, we modify the basis expansion method proposed by Zhang et al. and combine it with the improved Schnorr-Euchner search strategy to give a new KZ reduction algorithm. Simulation results show that the new KZ reduction algorithm is much faster and numerically more stable than the KZ reduction algorithm proposed by Zhang et al., especially when the basis matrix is ill conditioned.

**Index Terms**—KZ reduction, shortest vector problem, Schnorr-Euchner search algorithm, numerical stability.

## I. INTRODUCTION

Given a full column rank matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$ , the lattice  $\mathcal{L}(\mathbf{A})$  generated by  $\mathbf{A}$  is defined by

$$\mathcal{L}(\mathbf{A}) = \{\mathbf{A}\mathbf{x} \mid \mathbf{x} \in \mathbb{Z}^n\}. \quad (1)$$

The columns of  $\mathbf{A}$  form a basis of  $\mathcal{L}(\mathbf{A})$ . For any  $n \geq 2$ ,  $\mathcal{L}(\mathbf{A})$  has infinity many bases and any of two are connected by a unimodular matrix  $\mathbf{Z}$ , i.e.,  $\mathbf{Z} \in \mathbb{Z}^{n \times n}$  satisfies  $\det(\mathbf{Z}) = \pm 1$ . More precisely, for each given lattice basis matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$ ,  $\mathbf{AZ}$  is also a basis matrix of  $\mathcal{L}(\mathbf{A})$  if and only if  $\mathbf{Z}$  is unimodular, see, e.g., [1].

The process of selecting a good basis for a given lattice, given some criterion, is called lattice reduction. In many applications, it is advantageous if the basis vectors are short and close to be orthogonal [1]. For more than a century, lattice reduction has been investigated by many people and several types of reductions have been proposed, including the KZ reduction [2], the Minkowski reduction [3], the LLL reduction [4] and Seysen's reduction [5] etc.

Lattice reduction plays a crucial role in many areas, such as communications (see, e.g., [6] [1] [7]), combinatorial optimization (see, e.g., [8]), GPS (see, e.g., [9]), cryptography (see, e.g., [10] [11] [12]), number theory (see, e.g., [13] [14]), etc. For more details, see the survey paper [7] and references therein. Often in these applications, a closest vector problem

(CVP) (also referred to as an integer least squares problem) or a shortest vector problem (SVP) is usually need to be solved:

$$\min_{\mathbf{x} \in \mathbb{Z}^n} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2, \quad (2)$$

$$\min_{\mathbf{x} \in \mathbb{Z}^n \setminus \{0\}} \|\mathbf{A}\mathbf{x}\|_2. \quad (3)$$

In communications, CVP and SVP are usually solved by a sphere decoder which consists of two steps. In the first step, a lattice reduction, such as the LLL reduction and KZ reduction, is often used to preprocess the problems by reducing  $\mathbf{A}$  or  $(\mathbf{A}^\dagger)^T$  (here  $\mathbf{A}^\dagger = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T$ , the Moore-Penrose generalized inverse of  $\mathbf{A}$ , is a basis matrix of the dual lattice of  $\mathcal{L}(\mathbf{A})$ ). Then, in the second step, a search algorithm, typically the Schnorr-Euchner search strategy [15], an improvement of the Fincke-Pohst search strategy [16], is used to enumerate the integer vector within a hyper sphere. The first step, which is also called a preprocessing step, is to make the second step faster (see, e.g., [17] [18]).

The often used LLL reduction can be computed in polynomial time under some conditions and has some nice properties, see, e.g., [17] [19] [20] for some latest results. In some communication applications, one needs to solve a sequence of CVPs, where  $\mathbf{y}$ 's are different, but  $\mathbf{A}$ 's are identical. In this case, instead of using the LLL reduction, one usually uses the KZ reduction to do reduction, since the second step of the sphere decoding, which usually dominates the whole computational costs, becomes more efficient, although the KZ reduction costs more than the LLL reduction. Recently the KZ reduction has been used in integer-forcing linear receiver design [21] [22]. The properties of KZ reduced matrices have been studied in [23].

There are various KZ reduction algorithms, see, e.g., [1], [24]–[26]. Recently, a more efficient algorithm was proposed in [27]. As in [1], the LLL-aided Schnorr-Euchner search algorithm [15] is used to solve the SVPs in [27]. But instead of using Kannan's basis expansion method, which was used in [25] and [1], it uses a new basis expansion method, which is more efficient. However, the algorithm has a serious numerical problem, which may not only produce a lattice basis which is not KZ reduced, but also make the reduction process slow.

In this paper, we propose a modified KZ reduction algorithm to address the drawbacks of the algorithm presented in [27]. Part results of this paper have been presented in a conference paper [28]. In the following, we summarize the main contributions of this paper and point out the main difference between this paper and [28].

- An improved Schnorr-Euchner search algorithm for solving SVPs is proposed. This was not given in [28].
- On the one hand, we give a simple example which was not presented in [28] to show that some entries of a

This work was presented in part at the IEEE International Symposium on Information Theory (ISIT 2015), Hongkong.

J. Wen is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton T6G 2V4, Canada (e-mail: jinming1@ualberta.ca).

X.-W. Chang is with The School of Computer Science, McGill University, Montreal, QC H3A 0E9, Canada (e-mail: chang@cs.mcgill.ca).

solution of a general SVP can be arbitrary large. On the other hand, we derive upper bounds on the magnitudes of the entries of any solution of an SVP, whose basis matrix is LLL reduced. These bounds are sharper than those given in [28], which did not give a proof due to the space limitation.

- Combining the improved Schnorr-Euchner search algorithm with the modified basis expansion method in our conference paper [28], we get an improved KZ reduction algorithm. Numerical results indicate that it can be much faster and more numerically reliable than that proposed in [27].

The rest of the paper is organized as follows. In Section II, we introduce the LLL and KZ reductions. In Section III, we propose an improved Schnorr-Euchner search algorithm for solving the SVP. An example showing the entries of a solution of a general SVP can be infinitely large and upper bounds on the entries of any solution of a SVP with LLL reduced lattice basis are given in Section IV. Our modified KZ reduction algorithm is presented in Section V. Some simulation results are given in Section VI to show efficiency and numerical reliability of our new algorithms. Finally, we summarize this paper in Section VII.

*Notation.* For  $\mathbf{x} \in \mathbb{R}^n$ , we use  $\lfloor \mathbf{x} \rfloor$  to denote its nearest integer vector, i.e., each entry of  $\mathbf{x}$  is rounded to its nearest integer (if there is a tie, the one with smaller magnitude is chosen). For a matrix  $\mathbf{A}$ , we use  $a_{ij}$  to denote its  $(i, j)$  entry and use  $\mathbf{A}_{i:j, k:\ell}$  to denote the submatrix containing elements with row indices from  $i$  to  $j$  and column indices from  $k$  to  $\ell$ . Let  $\mathbf{e}_k$  denote the  $k$ -th column of an identity matrix  $\mathbf{I}$ , whose dimension depends on the context. For  $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{m \times n}$ , we denote  $|\mathbf{A}| = (|a_{ij}|)$ . For two matrices  $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$ , the inequality  $\mathbf{A} \leq \mathbf{B}$  means  $a_{ij} \leq b_{ij}$  for all  $i$  and  $j$ . For a scalar  $x$ , we denote

$$\text{sgn}(x) = \begin{cases} 1, & x \geq 0, \\ -1, & x < 0. \end{cases} \quad (4)$$

## II. LLL AND KZ REDUCTIONS

In this section, we introduce the LLL and KZ reductions.

### A. LLL reduction

Let  $\mathbf{A}$  in (1) have the following QR factorization (note that QR factorization algorithms can be found in many references, see, e.g., [29])

$$\mathbf{A} = [\mathbf{Q}_1, \mathbf{Q}_2] \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}, \quad (5)$$

where  $[\mathbf{Q}_1, \mathbf{Q}_2] \in \mathbb{R}^{m \times m}$  is orthogonal and  $\mathbf{R} \in \mathbb{R}^{n \times n}$  is upper triangular. Then, the LLL reduction [4] reduces  $\mathbf{R}$  in (5) to  $\bar{\mathbf{R}}$  via the QRZ factrization

$$\bar{\mathbf{Q}}^T \mathbf{R} \mathbf{Z} = \bar{\mathbf{R}}, \quad (6)$$

where  $\bar{\mathbf{Q}} \in \mathbb{R}^{n \times n}$  is orthogonal,  $\mathbf{Z} \in \mathbb{Z}^{n \times n}$  is unimodular and  $\bar{\mathbf{R}} \in \mathbb{R}^{n \times n}$  is upper triangular and satisfies the conditions: for  $1 \leq i \leq j-1 \leq n-1$ ,

$$|\bar{r}_{ij}| \leq \frac{1}{2} |\bar{r}_{ii}|, \quad (7)$$

$$\delta \bar{r}_{ii}^2 \leq \bar{r}_{ii}^2 + \bar{r}_{i+1, i+1}^2, \quad (8)$$

where  $\delta$  is a parameter satisfying  $1/4 < \delta \leq 1$ . Then the matrix  $\mathbf{AZ}$  is said to be LLL reduced. The equations (7) and (8) are respectively referred as the size-reduced condition and the Lovász condition.

### B. KZ reduction

Similar to the LLL reduction, after the QR factorization of  $\mathbf{A}$  (see (5)), the KZ reduction reduces  $\mathbf{R}$  in (5) to  $\bar{\mathbf{R}}$  through (6), where  $\bar{\mathbf{R}}$  satisfies (7) and

$$|\bar{r}_{ii}| = \min_{\mathbf{x} \in \mathbb{Z}^{n-i+1} \setminus \{\mathbf{0}\}} \|\bar{\mathbf{R}}_{i:n, i:n} \mathbf{x}\|_2, \quad 1 \leq i \leq n. \quad (9)$$

Then  $\mathbf{AZ}$  is said to be KZ reduced. Note that if a matrix is KZ reduced, it must be LLL reduced for  $\delta = 1$ .

## III. AN IMPROVED SCHNORR-EUCHNER SEARCH ALGORITHM FOR SVP

As mentioned in the introduction, solving an SVP is needed in many applications including performing KZ reduction. In this section, we modify the Schnorr-Euchner search algorithm to solve the SVP.

After the LLL reduction (6), SVP (3) is then transformed to

$$\min_{\mathbf{z} \in \mathbb{Z}^n \setminus \{\mathbf{0}\}} \|\bar{\mathbf{R}} \mathbf{z}\|_2. \quad (10)$$

Let  $\mathbf{z}$  be a solution of the SVP (10), then  $\mathbf{Z}\mathbf{z}$  is a solution of the SVP (3), where  $\mathbf{Z}$  is a unimodular matrix satisfying (6).

### A. The Schnorr-Euchner algorithm

In this section, we describe the Schnorr-Euchner search algorithm to solve the general SVP (10) so that it will be easier to introduce our improved version. More details on this algorithm can be found in e.g., [1], [30]–[32].

Assume that the solution  $\mathbf{z}$  of (10) is within the hyper-ellipsoid defined by

$$\|\bar{\mathbf{R}} \mathbf{z}\|_2^2 < \beta^2, \quad (11)$$

where  $\beta$  is a given constant. Let

$$c_i = -\frac{1}{\bar{r}_{ii}} \sum_{j=i+1}^n \bar{r}_{ij} z_j, \quad 1 \leq i \leq n, \quad (12)$$

where  $\sum_{i=n+1}^n \cdot = 0$ . Then (11) can be rewritten as

$$\sum_{i=1}^n \bar{r}_{ii}^2 (z_i - c_i)^2 < \beta^2$$

which is equivalent to

$$\bar{r}_{kk}^2 (z_k - c_k)^2 < \beta^2 - \sum_{i=k+1}^n \bar{r}_{ii}^2 (z_i - c_i)^2 \quad (13)$$

for  $k = n - 1, \dots, 1$ , where  $k$  is called the level index.

At level  $k$  in the search tree, the Schnorr-Euchner search algorithm tries to choose a value for  $z_k$  based on (13). At this point, the values of  $z_n, z_{n-1}, \dots, z_{k+1}$  have been chosen and the values of  $z_{k-1}, z_{k-2}, \dots, z_1$  have not been chosen yet. We now describe the search procedure briefly. First, we set  $\beta = \infty$ , and for  $k = n, n-1, \dots, 1$ , we compute  $c_k$  by (12) and set  $z_k = \lfloor c_k \rfloor$  which implies  $z_k = 0$  and (13) holds. Then  $\mathbf{z} = \mathbf{0}$  is obtained. We now update this  $\mathbf{z}$  to try to get another nonzero point. Specifically, we set  $z_1$  as the next closest integer to  $c_1$  – there are actually two choices and we take  $z_1 = 1$ . Since  $\beta = \infty$ , (13) with  $k = 1$  clearly holds for the updated  $\mathbf{z}$ . Thus, the first nonzero integer point, i.e.,  $\mathbf{z} = \mathbf{e}_1$ , is found. Now we update  $\beta$  by setting  $\beta = \|\bar{\mathbf{R}}\mathbf{z}\|_2$ . To find the optimal solution, we try to update  $\mathbf{z}$ . Since the two sides of (13) with  $k = 1$  are now equal, we cannot update  $z_1$  only, because this would increase the left hand side and as a result, (13) with  $k = 1$  does not hold. Thus, we move to level 2 to try to update  $z_2$  by setting it as the next closest integer to  $c_2$ . If (13) with  $k = 2$  holds, we move down to level 1 to update  $z_1$  with new  $c_1$  computed via (12); otherwise we move up to level 3 to try to update  $z_3$ , and so on. Finally, at level  $n$ , when we fail to find an integer  $z_n$  such (13) with  $k = n$  holds, the search process terminates and the latest found  $\mathbf{z}$  is a solution for the SVP (10).

#### B. An Improved Schnorr-Euchner algorithm

In this subsection, we modify the Schnorr-Euchner search algorithm which is used for solving the SVP (10) to make it faster.

Note that if

$$|\bar{r}_{11}| \leq |\bar{r}_{ii}|, \quad 2 \leq i \leq n, \quad (14)$$

then  $\mathbf{e}_1$  is a solution to (10). In fact, if  $\mathbf{z}$  is a solution with  $z_{k+1:n} = 0$  and  $z_k \neq 0$  for some  $k$ , then

$$\|\bar{\mathbf{R}}\mathbf{z}\|_2 \geq |\bar{r}_{kk}z_k| \geq |\bar{r}_{kk}| \geq |\bar{r}_{11}| = \|\bar{\mathbf{R}}\mathbf{e}_1\|_2,$$

implying that  $\mathbf{e}_1$  is a solution. Thus, before the search starts, we check if (14) holds. If it holds, then  $\mathbf{e}_1$  is a solution. Note that the LLL reduction on  $\mathbf{R}$  will help to enlarge the chance that (14) holds.

Obviously if  $\mathbf{z}$  is a solution to (10), so is  $-\mathbf{z}$ . Thus, to speed up the search, we only need to search the candidates  $\mathbf{z}$  with  $z_n \geq 0$ . This has been observed and used in [33] for solving a shortest independent vector problem in integer-forcing MIMO receiver design. We can extend the idea further. In the search process, if the current candidate  $\mathbf{z}$  satisfies  $z_{k+1:n} = \mathbf{0}$ , then we only need to search  $z_k \geq 0$  at level  $k$ .

The above two observations lead to an improved Schnorr-Euchner algorithm described in Algorithm 1. In Sec. VI-A, we will give simulation results to illustrate the improvement.

#### IV. UPPER BOUNDS ON THE SOLUTION OF THE SVP

In this section, we first give a simple example to show that some entries of the solution to a general SVP can be arbitrarily large. Then, we will prove that when the basis matrix of an SVP is LLL reduced, the entries of the solutions are bounded.

#### Algorithm 1 An improved Schnorr-Euchner search algorithm

**Input:** A nonsingular upper triangular  $\bar{\mathbf{R}} \in \mathbb{R}^{n \times n}$ .

**Output:** A solution  $\mathbf{z}$  to the SVP (10)

- 1) If (14) holds,  $\mathbf{z} = \mathbf{e}_1$ , return.
- 2) (Initialization) Set  $k = n, \beta = +\infty$ .
- 3) Compute  $c_k$  via (12), set  $z_k = \lfloor c_k \rfloor$  and  $s_k = \text{sgn}(c_k - z_k)$  (see (4)).
- 4) (Main Step) If (13) does not hold, then go to Step 5; else if  $k > 1$ , set  $k = k - 1$  and go to Step 3; else, i.e.,  $k = 1$ , go to Step 6.
- 5) (Outside ellipsoid) If  $k = n$ , return; else, set  $k = k + 1$  and go to Step 7.
- 6) (A point is found) If  $\mathbf{z} = \mathbf{0}$ , take  $z_1 = 1$ . Set  $\beta = \|\bar{\mathbf{R}}\mathbf{z}\|_2$  and  $k = k + 1$ .
- 7) (Enumeration at level  $k$ ) If  $k = n$  or  $z_{k+1:n} = \mathbf{0}$ , set  $z_k = z_k + 1$ , else, set  $z_k = z_k + s_k$ ,  $s_k = -s_k - \text{sgn}(s_k)$ . Go to Step 4.

The bounds are not only interesting in theory, but also useful in analyzing the complexity of the basis expansion in the KZ reduction.

**Example 1.** Let  $\bar{\mathbf{R}} = \begin{bmatrix} M & M^2 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{n-2} \end{bmatrix}$  with  $1 < M \in \mathbb{Z}$ . Then, for any nonzero  $\mathbf{z} \in \mathbb{Z}^n$ ,

$$\bar{\mathbf{R}}\mathbf{z} = [Mz_1 + M^2z_2, z_2, z_3, \dots, z_n]^T.$$

It is easy to show that  $\|\bar{\mathbf{R}}\mathbf{z}\|_2 \geq 1$ . In fact, if  $z_{2:n} = \mathbf{0}$ , then  $z_1 \neq 0$  and  $\|\bar{\mathbf{R}}\mathbf{z}\|_2 = |Mz_1| \geq M > 1$ ; otherwise,  $\|\bar{\mathbf{R}}\mathbf{z}\|_2 \geq \|z_{2:n}\|_2 \geq 1$ . Take  $\mathbf{z} = [M, -1, 0, \dots, 0]^T$ , then  $\|\bar{\mathbf{R}}\mathbf{z}\|_2 = 1$ . Thus this  $\mathbf{z}$  is a solution to the SVP (10). Since  $M$  can be arbitrarily large, this  $\mathbf{z}$  is unbounded.

When  $\bar{\mathbf{R}}$  is LLL reduced, however, we can show that any solution to the SVP (10) is bounded. To do this, we need to introduce the following lemma.

**Lemma 1.** Let  $\bar{\mathbf{R}}$  be the upper triangular matrix in (6). Define  $\hat{\mathbf{R}} = \mathbf{D}^{-1}\bar{\mathbf{R}}$ , where  $\mathbf{D} = \text{diag}(\bar{r}_{11}, \dots, \bar{r}_{nn})$ . Let  $\mathbf{U} \in \mathbb{R}^{n \times n}$  be an upper triangular matrix with

$$u_{ij} = \begin{cases} 1, & i = j, \\ \frac{1}{2} \left(\frac{3}{2}\right)^{j-i-1}, & i < j. \end{cases} \quad (15)$$

Suppose that  $\bar{\mathbf{R}}$  is size reduced, i.e., (7) holds, then

$$|\hat{\mathbf{R}}^{-1}| \leq \mathbf{U}. \quad (16)$$

*Proof.* This lemma is essentially the same as [34, Lemma 2] and it is a special case of the result given in the proof of [35, Theorem 3.2], which was easily derived by using the two results given in [36, Sections 8.2 and 8.3]. For readability, we give a proof here, which is different from those given in [34] and [35].

From the definition of  $\hat{\mathbf{R}}$  and the assumption that  $\bar{\mathbf{R}}$  is size reduced, we observe that

$$\hat{r}_{ii} = 1, \quad |\hat{r}_{ij}| \leq \frac{1}{2}, \quad 1 \leq i < j \leq n. \quad (17)$$

Let  $\mathbf{T} = \hat{\mathbf{R}}^{-1}$ , then

$$t_{ii} = 1 = u_{ii}, \quad 1 \leq i \leq n.$$

For any fixed  $j$ , we show that  $|t_{ij}| \leq u_{ij}$  for  $i = j-1, \dots, 1$  by induction on  $i$ . First we consider the base case  $i = j-1$ . Since

$$0 = (\hat{\mathbf{R}}\mathbf{T})_{j-1,j} = \hat{r}_{j-1,j-1}t_{j-1,j} + \hat{r}_{j-1,j}t_{jj},$$

where  $\hat{r}_{j-1,j-1} = t_{jj} = 1$ , we have

$$t_{j-1,j} = -\hat{r}_{j-1,j}.$$

Then from (17) we conclude that

$$|t_{j-1,j}| \leq \frac{1}{2} = u_{j-1,j}.$$

Suppose  $|t_{kj}| \leq u_{kj}$  for any  $k \leq i+1$ . Note that

$$0 = (\hat{\mathbf{R}}\mathbf{T})_{ij} = \hat{r}_{ii}t_{ij} + \hat{r}_{i,i+1}t_{i+1,j} + \dots + \hat{r}_{ij}t_{jj}, \quad i < j.$$

Then

$$t_{ij} = -\hat{r}_{i,i+1}t_{i+1,j} - \dots - \hat{r}_{ij}t_{jj}.$$

Then, by the fact that  $|\hat{r}_{ik}| \leq \frac{1}{2}$  for  $k > i$ , we obtain

$$\begin{aligned} |t_{ij}| &\leq \frac{1}{2} \left( \sum_{k=i+1}^{j-1} |t_{kj}| + t_{jj} \right) \leq \frac{1}{2} \left( \sum_{k=i+1}^{j-1} \frac{1}{2} \left( \frac{3}{2} \right)^{j-k-1} + 1 \right) \\ &= \frac{1}{2} \left( \frac{3}{2} \right)^{j-i-1} = u_{ij}. \end{aligned}$$

Thus, we can conclude that (16) holds.  $\square$

Here we make a remark. As essentially noticed in [34] (see also [36, eq. (8.4)]), in (7), if  $r_{ij} = -\frac{1}{2}r_{ii}$ , then  $\hat{r}_{ij} = -\frac{1}{2}$  for  $1 \leq i < j \leq n$ , and it is easy to see from the proof of the Lemma that  $\hat{\mathbf{R}}^{-1} = \mathbf{U}$ , so the upper bound (16) is attainable.

With Lemma 1, we can prove the following theorem.

**Theorem 1.** *Let  $\mathbf{z} \in \mathbb{Z}^n$  be a solution of (10), where  $\bar{\mathbf{R}}$  is LLL reduced, then*

$$|z_i| \leq \sqrt{\frac{1 - 2\alpha^2 - \frac{1}{9}(\frac{3}{2}\alpha)^{2(n-i+1)}}{1 - (\frac{3}{2}\alpha)^2}} \alpha^{i-1}, \quad 1 \leq i \leq n \quad (18)$$

where

$$\alpha = \frac{2}{\sqrt{4\delta - 1}} \quad (19)$$

with  $\delta$  being the parameter in the LLL reduction (see (8)).

*Proof.* Since  $\bar{\mathbf{R}}$  is LLL reduced, by (7) and (8), we have

$$\delta \bar{r}_{ii}^2 \leq \bar{r}_{i,i+1}^2 + \bar{r}_{i+1,i+1}^2 \leq \frac{1}{4} \bar{r}_{ii}^2 + \bar{r}_{i+1,i+1}^2, \quad 1 \leq i \leq n-1.$$

Then with (19),

$$\left| \frac{\bar{r}_{ii}}{\bar{r}_{i+1,i+1}} \right| \leq \alpha, \quad 1 \leq i \leq n-1.$$

Therefore,

$$\left| \frac{\bar{r}_{11}}{\bar{r}_{ii}} \right| \leq \alpha^{i-1}, \quad 1 \leq i \leq n, \quad (20)$$

which will be used later.

Since  $\mathbf{z}$  is a solution of (10),

$$\|\bar{\mathbf{R}}\mathbf{z}\|_2 \leq \|\bar{\mathbf{R}}\mathbf{e}_1\|_2 = |\bar{r}_{11}|.$$

Notice that

$$\mathbf{z}_i = \mathbf{e}_i^T \mathbf{z} = \mathbf{e}_i^T \bar{\mathbf{R}}^{-1} \bar{\mathbf{R}}\mathbf{z} = \mathbf{e}_i^T \hat{\mathbf{R}}^{-1} \mathbf{D}^{-1} \bar{\mathbf{R}}\mathbf{z},$$

where  $\mathbf{D}$  and  $\hat{\mathbf{R}}$  are defined in Lemma 1. Then by the Cauchy-Schwarz inequality and Lemma 1, we have

$$|z_i| \leq \|\mathbf{e}_i^T \hat{\mathbf{R}}^{-1} \mathbf{D}^{-1}\|_2 \|\bar{\mathbf{R}}\mathbf{z}\|_2 \quad (21)$$

$$\begin{aligned} &\leq \|\mathbf{e}_i^T \hat{\mathbf{R}}^{-1} \mathbf{D}^{-1}\|_2 |\bar{r}_{11}| \\ &\leq \|\mathbf{e}_i^T \mathbf{U} \mathbf{D}^{-1} \bar{r}_{11}\|_2. \end{aligned} \quad (22)$$

Note that from (15) and (20),

$$|\mathbf{U} \mathbf{D}^{-1} \bar{r}_{11}| \leq \begin{bmatrix} 1 & \frac{1}{2}\alpha & \frac{1}{2}(\frac{3}{2})\alpha^2 & \dots & \frac{1}{2}(\frac{3}{2})^{n-2}\alpha^{n-1} \\ & \alpha & \frac{1}{2}\alpha^2 & \dots & \frac{1}{2}(\frac{3}{2})^{n-3}\alpha^{n-1} \\ & & \alpha^2 & \dots & \frac{1}{2}(\frac{3}{2})^{n-4}\alpha^{n-1} \\ & & & \ddots & \vdots \\ & & & & \alpha^{n-1} \end{bmatrix}. \quad (23)$$

Then from (22) and (23), we obtain

$$\begin{aligned} |z_i| &\leq \sqrt{(\alpha^{i-1})^2 + \sum_{j=i+1}^n \left( \frac{1}{2} \left( \frac{3}{2} \right)^{j-i-1} \alpha^{j-1} \right)^2} \\ &= \sqrt{\frac{1 - 2\alpha^2 - \frac{1}{9}(\frac{3}{2}\alpha)^{2(n-i+1)}}{1 - (\frac{3}{2}\alpha)^2}} \alpha^{i-1}, \end{aligned}$$

completing the proof.  $\square$

The above theorem shows that when the basis matrix  $\bar{\mathbf{R}}$  is LLL reduced, any solution to (10) is bounded and the bounds in (18) depends on only the LLL reduction parameter  $\delta$  and the dimension  $n$ . This is crucial for the complexity of the basis expansion algorithm (see Sec. V). We need to point out that the bounds in (18) are not attainable. In fact, the equality in (21) holds if and only if  $\mathbf{D}^{-1} \hat{\mathbf{R}}^{-T} \mathbf{e}_i$  and  $\bar{\mathbf{R}}\mathbf{z}$  are linearly dependent, which is impossible for all  $i$  as  $\bar{\mathbf{R}}\mathbf{z} \neq \mathbf{0}$ .

## V. AN IMPROVED KZ REDUCTION ALGORITHM

In this section, we first introduce the KZ reduction algorithm given in [27], then propose an improved algorithm which is much faster and more numerical stable than this algorithm, especially when the basis matrix is ill conditioned.

### A. The KZ reduction algorithm in [27]

From the definition of the KZ reduction, the reduced matrix  $\bar{\mathbf{R}}$  satisfies both (7) and (9). If  $\bar{\mathbf{R}}$  in the QRZ factorization (6) satisfies (9), then we can easily apply size reductions to  $\bar{\mathbf{R}}$  such that (7) holds. Thus, in the following, we will only show how to obtain  $\bar{\mathbf{R}}$  such that (9) holds.

The algorithm needs  $n-1$  steps. Suppose that at the end of step  $k-1$ , one has found an orthogonal matrix  $\mathbf{Q}^{(k-1)} \in \mathbb{R}^{n \times n}$ , a unimodular matrix  $\mathbf{Z}^{(k-1)} \in \mathbb{Z}^{n \times n}$  and an upper triangular  $\mathbf{R}^{(k-1)} \in \mathbb{R}^{n \times n}$  such that

$$(\mathbf{Q}^{(k-1)})^T \mathbf{R} \mathbf{Z}^{(k-1)} = \mathbf{R}^{(k-1)}, \quad (24)$$

where for  $i = 1, \dots, k-1$ ,

$$|r_{ii}^{(k-1)}| = \min_{\mathbf{x} \in \mathbb{Z}^{n-i+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{i:n,i:n}^{(k-1)} \mathbf{x}\|_2. \quad (25)$$

At step  $k$ , like [1], [27] uses the LLL-aided Schnorr-Euchner search algorithm to solve the SVP:

$$\mathbf{x}^{(k)} = \arg \min_{\mathbf{x} \in \mathbb{Z}^{n-k+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x}\|_2^2. \quad (26)$$

Then, unlike the KZ reduction algorithms in [1], [24]–[26], [27] finds the unimodular matrix by expanding  $\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x}^{(k)}$  to a basis for the lattice  $\{\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x} : \mathbf{x} \in \mathbb{Z}^{n-k+1}\}$ . Specifically, [27] first constructs a unimodular matrix  $\tilde{\mathbf{Z}}^{(k)} \in \mathbb{Z}^{(n-k+1) \times (n-k+1)}$  whose first column is  $\mathbf{x}^{(k)}$ , i.e.,

$$\tilde{\mathbf{Z}}^{(k)} \mathbf{e}_1 = \mathbf{x}^{(k)}, \quad (27)$$

and then finds an orthogonal matrix  $\tilde{\mathbf{Q}}^{(k)} \in \mathbb{R}^{(n-k+1) \times (n-k+1)}$  to bring  $\mathbf{R}_{k:n,k:n}^{(k-1)} \tilde{\mathbf{Z}}^{(k)}$  back to an upper triangular matrix  $\tilde{\mathbf{R}}^{(k)}$ , i.e.,

$$(\tilde{\mathbf{Q}}^{(k)})^T \mathbf{R}_{k:n,k:n}^{(k-1)} \tilde{\mathbf{Z}}^{(k)} = \tilde{\mathbf{R}}^{(k)}. \quad (28)$$

Based on (24) and (28), we define

$$\mathbf{Q}^{(k)} = \mathbf{Q}^{(k-1)} \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{Q}}^{(k)} \end{bmatrix}, \quad (29)$$

$$\mathbf{R}^{(k)} = \begin{bmatrix} \mathbf{R}_{1:k-1,1:k-1}^{(k-1)} & \mathbf{R}_{1:k-1,k:n}^{(k-1)} \tilde{\mathbf{Z}}^{(k)} \\ \mathbf{0} & \tilde{\mathbf{R}}^{(k)} \end{bmatrix}, \quad (30)$$

$$\mathbf{Z}^{(k)} = \mathbf{Z}^{(k-1)} \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{Z}}^{(k)} \end{bmatrix}. \quad (31)$$

Here  $\mathbf{Q}^{(k)}$  is orthogonal,  $\mathbf{R}^{(k)}$  is upper triangular and  $\mathbf{Z}^{(k)}$  is unimodular. Then, combining (24) and (28), we obtain

$$(\mathbf{Q}^{(k)})^T \mathbf{R} \mathbf{Z}^{(k)} = \mathbf{R}^{(k)}. \quad (32)$$

At the end of step  $n-1$ , we get  $\mathbf{R}^{(n-1)}$ , which is just  $\bar{\mathbf{R}}$  in (6). In the following we explain why (9) holds. To show this, we only need to show that (25) holds for  $i = 1, \dots, k$  when  $k-1$  changes to  $k$ .

From (30) and (28), it is easy to verify that for  $i = 1, \dots, k$ ,

$$\mathbf{R}_{i:n,i:n}^{(k)} = \begin{bmatrix} \mathbf{I}_{k-i} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{Q}}^{(k)} \end{bmatrix}^T \mathbf{R}_{i:n,i:n}^{(k-1)} \begin{bmatrix} \mathbf{I}_{k-i} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{Z}}^{(k)} \end{bmatrix}. \quad (33)$$

Then, from (33) and (25), for  $i = 1, \dots, k-1$ ,

$$\begin{aligned} |r_{ii}^{(k)}| &= |r_{ii}^{(k-1)}| = \min_{\mathbf{x} \in \mathbb{Z}^{n-i+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{i:n,i:n}^{(k-1)} \mathbf{x}\|_2 \\ &= \min_{\mathbf{z} \in \mathbb{Z}^{n-i+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{i:n,i:n}^{(k)} \mathbf{z}\|_2, \end{aligned}$$

where

$$\mathbf{z} = \begin{bmatrix} \mathbf{I}_{k-i} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{Z}}^{(k)} \end{bmatrix}^{-1} \mathbf{x}. \quad (34)$$

Thus, we obtain

$$\begin{aligned} |r_{kk}^{(k)}| &= \|\tilde{\mathbf{R}}^{(k)} \mathbf{e}_1\| \\ &\stackrel{(a)}{=} \|(\tilde{\mathbf{Q}}^{(k)})^T \mathbf{R}_{k:n,k:n}^{(k-1)} \tilde{\mathbf{Z}}^{(k)} \mathbf{e}_1\| \\ &\stackrel{(b)}{=} \|\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x}^{(k)}\| \\ &\stackrel{(c)}{=} \min_{\mathbf{x} \in \mathbb{Z}^{n-k+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x}\|_2 \\ &\stackrel{(d)}{=} \min_{\mathbf{x} \in \mathbb{Z}^{n-k+1} \setminus \{\mathbf{0}\}} \|\tilde{\mathbf{R}}^{(k)} (\tilde{\mathbf{Z}}^{(k)})^{-1} \mathbf{x}\|_2 \\ &\stackrel{(e)}{=} \min_{\mathbf{z} \in \mathbb{Z}^{n-k+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{k:n,k:n}^{(k)} \mathbf{z}\|_2, \end{aligned}$$

where (a) and (d) are from (28), (b) is due to (27) and the fact that  $\tilde{\mathbf{Q}}^{(k)}$  is an orthogonal matrix, (c) follows from (26) and (e) is from (30) and (34). Thus (25) holds when  $k-1$  changes to  $k$ . Then, with  $\bar{\mathbf{R}} = \mathbf{R}^{(n-1)}$ , we can conclude (9) holds.

In the following, we introduce the process of obtaining the unimodular matrix  $\tilde{\mathbf{Z}}^{(k)}$  in (27) proposed in [27]. (There are some other methods to find  $\tilde{\mathbf{Z}}^{(k)}$ , see, e.g., [37, pp.13].) Suppose that  $\mathbf{z} = [p, q]^T \in \mathbb{Z}^2$  and  $\gcd(p, q) = d$ , then, there exist two integers  $a$  and  $b$  such that  $ap + bq = d$ . The extended Euclid algorithm can be used to find  $a$ ,  $b$  and  $d$  [38, p.325]. Obviously,

$$\mathbf{U} = \begin{bmatrix} p/d & -b \\ q/d & a \end{bmatrix} \quad (35)$$

is unimodular and it is easy to verify that  $\mathbf{U}^{-1} \mathbf{z} = d \mathbf{e}_1$ .

From (26), we can conclude that

$$\gcd(x_1^{(k)}, x_2^{(k)}, \dots, x_{n-k+1}^{(k)}) = 1.$$

After  $\mathbf{x}^{(k)}$  is obtained,  $\tilde{\mathbf{Z}}^{(k)}$  can be obtained by applying a sequence of 2 by 2 unimodular transformations of the form (35) to transform  $\mathbf{x}^{(k)}$  to  $\mathbf{e}_1$ , i.e.,  $(\tilde{\mathbf{Z}}^{(k)})^{-1} \mathbf{x}^{(k)} = \mathbf{e}_1$  (see (27)). Specifically, they eliminate the entries of  $\mathbf{x}^{(k)}$  from the last one to the second one. The resulting algorithm for finding  $\tilde{\mathbf{Z}}^{(k)}$  is described in Algorithm 2 and the corresponding KZ reduction algorithm is described in Algorithm 3.

Here we make a remark. Algorithm 3 does not show how to form and update  $\mathbf{Q}$ , as it may not be needed in applications. If an application indeed needs  $\mathbf{Q}$ , then we can obtain it by the QR factorization of  $\mathbf{AZ}$  after obtaining  $\mathbf{Z}$ . This would be more efficient.

### B. An improved KZ reduction algorithm

In this subsection, we first modify Algorithm 2 and then modify Algorithm 3 to get a new KZ reduction algorithm, which can be much faster and more numerically reliable than Algorithm 3.

First, we make an observation on Algorithm 3 and make a simple modification. At step  $k$ , if  $\mathbf{x}^{(k)} = \pm \mathbf{e}_1$  (see (26)), then, obviously, the basis expansion step, i.e., Algorithm 2, is not needed and we can move to step  $k+1$ . Later we will come back to this observation again.

In the following, we will make some major modifications. But before doing it, we introduce the following basic fact: For

---

**Algorithm 2** The Basis Expansion Algorithm in [27]

---

**Input:** An upper triangular  $\mathbf{R} \in \mathbb{R}^{n \times n}$ , a unimodular  $\mathbf{Z} \in \mathbb{Z}^{n \times n}$ , the index  $k$  and  $\mathbf{x} \in \mathbb{Z}^{n-k+1}$ , a solution to the SVP  $\min_{\mathbf{x} \in \mathbb{Z}^{n-k+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{k:n,k:n} \mathbf{x}\|_2$ .

**Output:** The updated upper triangular  $\mathbf{R}$  with  $r_{kk} = \|\mathbf{R}_{k:n,k:n}\|_2$  and the updated unimodular matrix  $\mathbf{Z}$ .

- 1: **for**  $i = n - k, \dots, 1$  **do**
- 2:   find  $d = \gcd(x_i, x_{i+1})$  and integers  $a$  and  $b$  such that  $ax_i + bx_{i+1} = d$ ;
- 3:   set  $\mathbf{U} = \begin{bmatrix} x_i/d & -b \\ x_{i+1}/d & a \end{bmatrix}$ ;  $x_i = d$ ;
- 4:    $\mathbf{Z}_{1:n,k+i-1:k+i} = \mathbf{Z}_{1:n,k+i-1:k+i} \mathbf{U}$ ;
- 5:    $\mathbf{R}_{1:k+i,k+i-1:k+i} = \mathbf{R}_{1:k+i,k+i-1:k+i} \mathbf{U}$ ;
- 6:   find a  $2 \times 2$  Givens rotation  $\mathbf{G}$  such that:

$$\mathbf{G} \begin{bmatrix} r_{k+i-1,k+i-1} \\ r_{k+i,k+i-1} \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix};$$

- 7:    $\mathbf{R}_{k+i-1:k+i,k+i-1:n} = \mathbf{G} \mathbf{R}_{k+i-1:k+i,k+i-1:n}$ ;
  - 8: **end for**
- 

---

**Algorithm 3** The KZ Reduction Algorithm in [27]

---

**Input:** A full column rank matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$

**Output:** A KZ reduced upper triangular  $\mathbf{R} \in \mathbb{R}^{n \times n}$  and the corresponding unimodular matrix  $\mathbf{Z} \in \mathbb{Z}^{n \times n}$ .

- 1: compute the QR factorization of  $\mathbf{A}$ , see (5);
  - 2: set  $\mathbf{Z} = \mathbf{I}$ ;
  - 3: **for**  $k = 1$  to  $n - 1$  **do**
  - 4:   solve  $\min_{\mathbf{x} \in \mathbb{Z}^{n-k+1} \setminus \{\mathbf{0}\}} \|\mathbf{R}_{k:n,k:n} \mathbf{x}\|_2$  by the LLL-aided Schnorr-Euchner search strategy;
  - 5:   apply Algorithm 2 to update  $\mathbf{R}$  and  $\mathbf{Z}$ ;
  - 6: **end for**
  - 7: perform size reductions on  $\mathbf{R}$  and update  $\mathbf{Z}$
- 

any two integers  $p$  and  $q$ , the time complexity of finding two integers  $a$  and  $b$  such that  $ap + bq = d \equiv \gcd(p, q)$  by the extended Euclid algorithm is bounded by  $\mathcal{O}(\log_2(\min\{|p|, |q|\}))$  if fixed precision is used [39].

In Algorithm 3, after finding  $\mathbf{x}^{(k)}$  (see (26)), Algorithm 2 is used to expand  $\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x}^{(k)}$  to a basis for the lattice  $\{\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x} : \mathbf{x} \in \mathbb{Z}^{n-k+1}\}$ . There are some serious drawbacks with this approach.

- Sometimes, especially when  $\mathbf{A}$  is ill-conditioned, some of the entries of  $\mathbf{x}^{(k)}$  may be very large such that they are beyond the range of consecutive integers in a floating point system (i.e., integer overflow occurs), very likely resulting in wrong results. Even if integer overflow does not occur in storing  $\mathbf{x}^{(k)}$ , large  $\mathbf{x}^{(k)}$  may still cause problems. One problem is that the computational cost of the extended Euclid algorithm will be high according to its complexity result we just mentioned before.
- The second problem is that updating  $\mathbf{Z}$  and  $\mathbf{R}$  in lines 4 and 5 of Algorithm 2 may cause numerical issues. Large  $x_i$  and  $x_{i+1}$  are likely to produce large elements in  $\mathbf{U}$ . As a result, integer overflow may occur in updating  $\mathbf{Z}$ , and large rounding errors are likely to occur in updating

$\mathbf{R}$ .

- Finally,  $\mathbf{R}$  is likely to become more ill-conditioned after the updating, making the search process for solving SVPs in later steps expensive.

In order to deal with the large  $\mathbf{x}^{(k)}$  issue, we look at line 4 in Algorithm 3, which uses the LLL-aided Schnorr-Euchner search algorithm to solve the SVP. Specifically at step  $k$ , to solve (26), the LLL reduction algorithm is applied to  $\mathbf{R}_{k:n,k:n}^{(k-1)}$ :

$$(\hat{\mathbf{Q}}^{(k)})^T \mathbf{R}_{k:n,k:n}^{(k-1)} \hat{\mathbf{Z}}^{(k)} = \hat{\mathbf{R}}^{(k-1)} \quad (36)$$

where  $\hat{\mathbf{Q}}^{(k)} \in \mathbb{R}^{(n-k+1) \times (n-k+1)}$  is orthogonal,  $\hat{\mathbf{Z}}^{(k)} \in \mathbb{Z}^{(n-k+1) \times (n-k+1)}$  is unimodular and  $\hat{\mathbf{R}}^{(k-1)}$  is LLL-reduced. Then, one solves the reduced SVP:

$$\mathbf{z}^{(k)} = \arg \min_{\mathbf{z} \in \mathbb{Z}^{n-k+1} \setminus \{\mathbf{0}\}} \|\hat{\mathbf{R}}^{(k-1)} \mathbf{z}\|_2^2. \quad (37)$$

The solution of the original SVP is  $\mathbf{x}^{(k)} = \hat{\mathbf{Z}}^{(k)} \mathbf{z}^{(k)}$ . We will use the improved Schnorr-Euchner search algorithm (Algorithm 1) to solve the SVPs.

Instead of expanding  $\mathbf{R}_{k:n,k:n}^{(k-1)} \mathbf{x}^{(k)}$  as done in Algorithm 3, we propose to expand  $\hat{\mathbf{R}}^{(k-1)} \mathbf{z}^{(k)}$  to a basis for the lattice  $\{\hat{\mathbf{R}}^{(k-1)} \mathbf{z} : \mathbf{z} \in \mathbb{Z}^{n-k+1}\}$ . Unlike  $\mathbf{x}^{(k)}$  in (26), which can be arbitrarily large,  $\mathbf{z}^{(k)}$  in (37) is bounded (see Theorem 1).

Thus, before doing the expansion, we update  $\mathbf{Q}^{(k)}$ ,  $\mathbf{R}^{(k)}$  and  $\mathbf{Z}^{(k)}$  by using the LLL reduction (36):

$$\check{\mathbf{Q}}^{(k)} = \mathbf{Q}^{(k-1)} \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \hat{\mathbf{Q}}^{(k)} \end{bmatrix}, \quad (38)$$

$$\check{\mathbf{R}}^{(k)} = \begin{bmatrix} \mathbf{R}_{1:k-1,1:k-1}^{(k-1)} & \mathbf{R}_{1:k-1,k:n}^{(k-1)} \hat{\mathbf{Z}}^{(k)} \\ \mathbf{0} & \hat{\mathbf{R}}^{(k-1)} \end{bmatrix}, \quad (39)$$

$$\check{\mathbf{Z}}^{(k)} = \mathbf{Z}^{(k-1)} \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \hat{\mathbf{Z}}^{(k)} \end{bmatrix}. \quad (40)$$

Now we do expansion. We construct a unimodular matrix  $\tilde{\mathbf{Z}}^{(k)} \in \mathbb{Z}^{(n-k+1) \times (n-k+1)}$  whose first column is  $\mathbf{z}^{(k)}$ , and find an orthogonal matrix  $\tilde{\mathbf{Q}}^{(k)}$  to bring  $\hat{\mathbf{R}}^{(k-1)} \tilde{\mathbf{Z}}^{(k)}$  back to an upper triangular matrix  $\tilde{\mathbf{R}}^{(k)}$  (cf. (28)):

$$(\tilde{\mathbf{Q}}^{(k)})^T \hat{\mathbf{R}}^{(k-1)} \tilde{\mathbf{Z}}^{(k)} = \tilde{\mathbf{R}}^{(k)}. \quad (41)$$

Then, we update  $\check{\mathbf{Q}}^{(k)}$ ,  $\check{\mathbf{R}}^{(k)}$  and  $\check{\mathbf{Z}}^{(k)}$  as follows (cf. (29)–(31)):

$$\mathbf{Q}^{(k)} = \check{\mathbf{Q}}^{(k-1)} \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{Q}}^{(k)} \end{bmatrix}, \quad (42)$$

$$\mathbf{R}^{(k)} = \begin{bmatrix} \mathbf{R}_{1:k-1,1:k-1}^{(k-1)} & \mathbf{R}_{1:k-1,k:n}^{(k-1)} \tilde{\mathbf{Z}}^{(k)} \\ \mathbf{0} & \tilde{\mathbf{R}}^{(k)} \end{bmatrix}, \quad (43)$$

$$\mathbf{Z}^{(k)} = \check{\mathbf{Z}}^{(k-1)} \begin{bmatrix} \mathbf{I}_{k-1} & \mathbf{0} \\ \mathbf{0} & \tilde{\mathbf{Z}}^{(k)} \end{bmatrix} \quad (44)$$

and we obtain the QRZ factorization of  $\mathbf{R}$  in the same form as (32) at step  $k$ .

Now we discuss the advantages of our modifications.

- First, the improved Schnorr-Euchner search strategy algorithm is more efficient than the original one.
- Second, we expand  $\hat{\mathbf{R}}^{(k-1)} \mathbf{z}^{(k)}$  to a basis for the lattice  $\{\hat{\mathbf{R}}^{(k-1)} \mathbf{z} : \mathbf{z} \in \mathbb{Z}^{n-k+1}\}$ , and do not transfer  $\mathbf{z}^{(k)}$  back to  $\mathbf{x}^{(k)}$  as Algorithm 3 does, i.e., we do not need

to compute  $\mathbf{x}^{(k)} = \widehat{\mathbf{Z}}^{(k)} \mathbf{z}^{(k)}$  which can save some computational cost.

- Third, since  $\widehat{\mathbf{R}}^{(k-1)}$  is LLL reduced, it has a very good chance, especially when  $\mathbf{R}$  is well-conditioned and  $n$  is small (say, smaller than 30), that  $\mathbf{z}^{(k)} = \pm \mathbf{e}_1$  (see (37)). This was observed in our simulations. As we stated before, the basis expansion is not needed in this case and we can move to next step.
- Finally, the entries of  $\mathbf{z}^{(k)}$  are bounded according to Theorem 1, but the entries of  $\mathbf{x}^{(k)}$  may not be bounded (see Example 1). Our simulations indicated that the former are smaller or much smaller than the latter. Thus, the serious problems with using  $\mathbf{x}^{(k)}$  for basis expansion mentioned before can be significantly mitigated by using  $\mathbf{z}^{(k)}$  instead.

Now look at Algorithm 2. This basis expansion algorithm finds a sequence of 2 by 2 unimodular matrices in the form of (35) to eliminate the entries of  $\mathbf{x}$  from the last one to the second one. Note that for any fixed  $i$  (see line 1), if  $x_{i+1} = 0$ , lines 2-7 do not need to be performed and we only need to move to the next iteration. In our simulations we noticed that  $\mathbf{z}^{(k)}$  (see (37)) often has a lot of zeros. and the above modification to the basis expansion algorithm can reduce the computational cost.

Based on the above discussion, we now present an improved KZ reduction algorithm in Algorithm 4.

---

**Algorithm 4** An Improved KZ Reduction Algorithm

---

**Input:** A full column rank matrix  $\mathbf{A} \in \mathbb{R}^{m \times n}$

**Output:** A KZ reduced upper triangular  $\mathbf{R} \in \mathbb{R}^{n \times n}$  and the corresponding unimodular matrix  $\mathbf{Z} \in \mathbb{Z}^{n \times n}$ .

---

- 1: compute the QR factorization of  $\mathbf{A}$ , see (5);
  - 2: set  $\mathbf{Z} = \mathbf{I}, k = 1$ ;
  - 3: **while**  $k < n$  **do**
  - 4:   compute the LLL reduction of  $\mathbf{R}_{k:n, k:n}$  (see (36)) and update  $\mathbf{R}, \mathbf{Z}$  (see (39)-(40));
  - 5:   solve  $\min_{\mathbf{z} \in \mathbb{Z}^{n-k+1} \setminus \{0\}} \|\mathbf{R}_{k:n, k:n} \mathbf{z}\|_2^2$  by Algorithm 1 to get the solution  $\mathbf{z}$ ;
  - 6:   **if**  $\mathbf{z} = \mathbf{e}_1$  **then**
  - 7:      $k = k + 1$ ;
  - 8:   **else**
  - 9:      $i = n - k$ ;
  - 10:    **while**  $i \geq 1$  **do**
  - 11:     **if**  $z_{i+1} \neq 0$  **then**
  - 12:      perform lines 2-7 of Algorithm 2 (where  $x_i$  and  $x_{i+1}$  are replaced by  $z_i$  and  $z_{i+1}$ );
  - 13:     **end if**
  - 14:      $i = i - 1$ ;
  - 15:    **end while**
  - 16:     $k = k + 1$ ;
  - 17:   **end if**
  - 18: **end while**
  - 19: perform size reductions on  $\mathbf{R}$  and update  $\mathbf{Z}$ .
- 

### C. A concrete example

As stated in the above subsection, Algorithm 3 has numerical issues. In this subsection, we give an example to show that Algorithm 3 may not even give a LLL reduced matrix (for  $\delta = 1$ ), while Algorithm 4 does.

**Example 2.** Let

$$\mathbf{A} = \begin{bmatrix} 10.6347 & -66.2715 & 9.3046 & 17.5349 & 24.9625 \\ 0 & 8.6759 & -4.7536 & -3.9379 & -2.3318 \\ 0 & 0 & 0.3876 & 0.1296 & -0.2879 \\ 0 & 0 & 0 & 0.0133 & -0.0082 \\ 0 & 0 & 0 & 0 & 0.0015 \end{bmatrix}.$$

Applying Algorithm 3 gives

$$\mathbf{R} = \begin{bmatrix} -0.2256 & -0.0792 & 0.0125 & 0 & 0 \\ 0 & 0.2148 & -0.0728 & -0.0029 & -0.0012 \\ 0 & 0 & 0.2145 & 0.0527 & -0.0211 \\ 0 & 0 & 0 & -0.1103 & 0.0306 \\ 0 & 0 & 0 & 0 & 0.6221 \end{bmatrix}.$$

It is easy to check that  $\mathbf{R}$  is not LLL reduced (for  $\delta = 1$ ). In fact,  $r_{33}^2 > r_{34}^2 + r_{44}^2$ . Moreover, the matrix  $\mathbf{Z}$  obtained by Algorithm 3 is not unimodular since its determinant is  $-3244032$ , which was precisely calculated by Maple. The reason for this is that  $\mathbf{A}$  is ill conditioned (its condition number in the 2-norm is about  $1.0 \times 10^5$ ) and some of the entries of  $\mathbf{x}^{(k)}$  (see (26)) are too large, causing severe inaccuracy in updating  $\mathbf{R}$  and integer overflow in updating  $\mathbf{Z}$  (see lines 4-5 in Algorithm 2). In fact,

$$\begin{aligned} \mathbf{x}^{(1)} &= [-47, -27, -21, -14, -34]^T; \\ \mathbf{x}^{(2)} &= [-48029, -27593, 2145, 345]^T; \\ \mathbf{x}^{(3)} &= [-2767925153, 432235, 40]^T; \\ \mathbf{x}^{(4)} &= [691989751, 2]^T. \end{aligned}$$

The condition numbers in the 2-norm of  $\mathbf{R}(k:5, k:5)$  obtained at the end of step  $k = 1, 2, 3, 4$  of Algorithm 3 are respectively  $2.9 \times 10^8, 1.5 \times 10^{15}, 6.2 \times 10^{18}$  and  $1.1 \times 10$ . Since the computed  $\mathbf{R}$  accumulates errors in the reduction process and it may be far away from the true  $\mathbf{R}$ -factor of  $\mathbf{AZ}$ , one may suspect that  $\mathbf{AZ}$  is still LLL reduced. Actually we found it is still not by looking at the  $\mathbf{R}$ -factor of the QR factorization of  $\mathbf{AZ}$ .

Applying Algorithm 4 to  $\mathbf{A}$  gives

$$\mathbf{R} = \begin{bmatrix} -0.2256 & 0.0792 & -0.0126 & 0.0028 & -0.0621 \\ 0 & -0.2148 & 0.0728 & -0.0084 & 0.0930 \\ 0 & 0 & 0.2145 & 0.0292 & -0.0029 \\ 0 & 0 & 0 & -0.2320 & 0.0731 \\ 0 & 0 & 0 & 0 & -0.2959 \end{bmatrix}.$$

Although we cannot verify whether  $\mathbf{R}$  is KZ reduced or not, we can verify that indeed it is LLL reduced. All of the solutions of the four SVPs are  $\mathbf{e}_1$  (note that the dimensions are different). Thus, no basis expansion is needed. The condition numbers in the 2-norm of  $\mathbf{R}(k:5, k:5)$  obtained at the end of step  $k = 1, 2, 3, 4$  of Algorithm 4 are respectively 2.1, 1.9, 1.6 and 1.4.

## VI. NUMERICAL TESTS

In this section, we compare the performances of the modified Schnorr-Euchner search algorithm, i.e., Algorithm 1, with the original one, i.e., the one proposed in [15], and compare the modified KZ reduction algorithm, i.e., Algorithm 4, with that in [27], i.e., Algorithm 3. All the numerical tests were done by MATLAB 2016b on a desktop computer with Intel(R) Xeon(R) CPU E5-1603 v4 @ 2.8GHz. The MATLAB code for Algorithm 3 was provided by Dr. Wen Zhang, one of the authors of [27]. The parameter  $\delta$  in the LLL reduction was chosen to be 1.

We compare the efficiencies of the algorithms in terms of the CPU time by doing tests for the following two classes of matrices.

- Case 1.  $\mathbf{A} = \text{randn}(n)$ , where  $\text{randn}(n)$  is a MATLAB built-in function which generates a random  $n \times n$  matrix, whose entries independent and identically follow the standard normal distribution  $\mathcal{N}(0, 1)$ .
- Case 2.  $\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T$ , where  $\mathbf{U}$  and  $\mathbf{V}$  are the Q-factors of the QR factorization of random matrices generated by  $\text{randn}(n)$  and  $\mathbf{D}$  is a  $n \times n$  diagonal matrix with  $d_{ii} = 10^{3(n/2-i)/(n-1)}$ . Thus the condition number of  $\mathbf{A}$  is 1000.

### A. Comparison of the Search Strategies

In this subsection, we compare the efficiencies of Algorithm 1 which is the improved Schnorr-Euchner search algorithm, with the original Schnorr-Euchner search algorithm in [15]. In the tests for each case for a fixed  $n$  we gave 200 runs to generate 200 different  $\mathbf{A}$ 's. Figures 1 and 2 display the average CPU time of these two search algorithms over 200 runs versus  $n = 2 : 2 : 20$  for Cases 1 and 2, respectively. In both figures, "SE" and "Improved SE" refer to the original Schnorr-Euchner search algorithm and Algorithm 1, respectively.

From Figures 1 and 2, we can see that Algorithm 1 is faster than the one in [15].

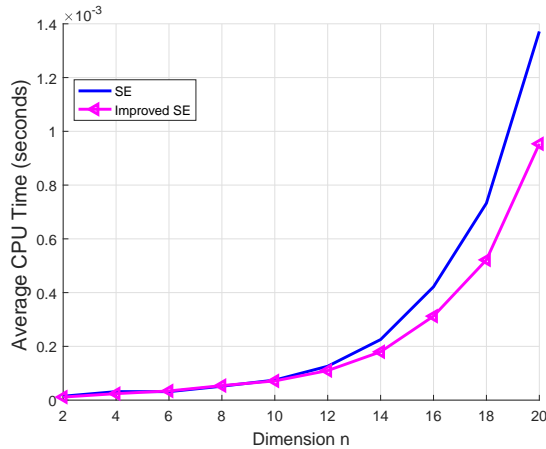


Fig. 1. Average CPU time of original and improved Schnorr-Euchner search algorithms versus  $n$  for Case 1

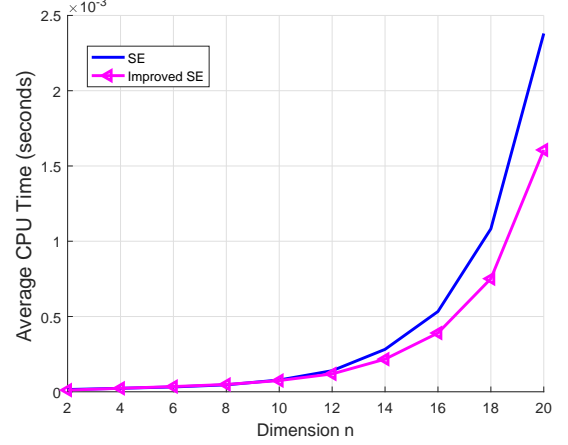


Fig. 2. Average CPU time of original and improved Schnorr-Euchner search algorithms versus  $n$  for Case 2

### B. Comparison of the KZ reduction algorithms

In this subsection, we compare the efficiencies of Algorithm 4 which is the improved KZ reduction algorithm, with Algorithm 3 which is the KZ reduction algorithm in [27]. In the numerical tests for each case for a fixed  $n$  we gave 200 runs to generate 200 different  $\mathbf{A}$ 's. Figures 3 and 4 display the average CPU time of these two KZ reduction algorithms over 200 runs versus  $n = 2 : 2 : 20$  for Cases 1 and 2, respectively. In both figures, "KZ" and "Improved KZ" refer to Algorithms 3 and 4, respectively.

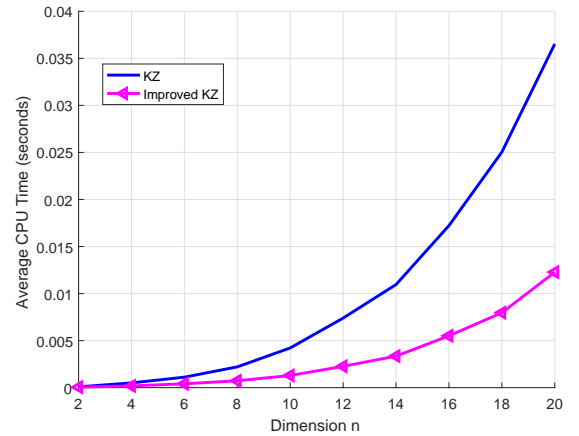


Fig. 3. Average CPU time of KZ reduction algorithms versus  $n$  for Case 1

Figure 4 gives the results for only  $n = 2 : 2 : 10$  for Algorithm 3. This is because when  $n \geq 12$ , it often did not terminate within ten hours.

In Case 1, sometimes Algorithm 3 did not terminate within a half hour and we just ignored this instance and gave one more run. The number of such instances was much smaller than that for Case 2.

From Figures 3 and 4, we can see that Algorithm 4 is faster than Algorithm 3 for both the two cases, and the improvement for Case 2 is more significant than that for Case 1. Also, when we ran Algorithm 3 we got a warning message "Warning:



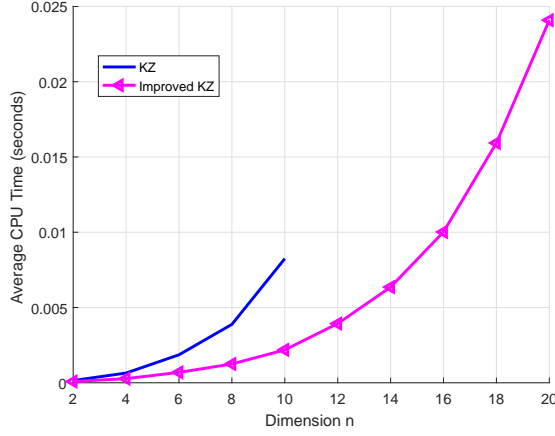


Fig. 4. Average CPU time of KZ reduction algorithms versus  $n$  for Case 2

Inputs contain values larger than the largest consecutive flint. Result may be inaccurate many times, for both Cases 1 and 2 in the tests. But this did not happen to Algorithm 4. Thus Algorithm 4 is more numerically reliable.

## VII. SUMMARY AND COMMENT

The KZ reduction has lot of applications including in communications and cryptography. In this paper, we investigated the KZ reduction and developed an KZ reduction algorithm. We first modified the Schnorr-Euchner search strategy for solving SVPs. Then, we investigated the magnitudes of the entries of the solutions of SVPs which are useful for analyzing the complexity of KZ reduction algorithms. On the one hand, a simple example which shows some entries of the solutions of a general SVP can be infinitely larger was given. On the other hand, we rigorously showed that the entries of the solutions of the SVPs whose bases matrices are LLL reduced can be upper bounded. Finally, we developed a novel KZ reduction algorithm by combining the improved Schnorr-Euchner search strategy with an improved basis expansion method. Simulation results showed that the modified Schnorr-Euchner search strategy for solving SVPs, and the new KZ reduction algorithm is much more efficient and more numerically stable than the one developed in [27] especially when the bases matrices are ill conditioned.

## REFERENCES

- [1] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, 2002.
- [2] A. Korkine and G. Zolotareff, "Sur les formes quadratiques," *Math. Ann.*, vol. 6, no. 3, pp. 366–389, 1873.
- [3] H. Minkowski, "Geometrie der zahlen (2 vol.)," *Teubner, Leipzig*, vol. 1910, 1896.
- [4] A. Lenstra, H. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients," *Math. Ann.*, vol. 261, no. 4, pp. 515–534, 1982.
- [5] M. Seysen, "Simultaneous reduction of a lattice basis and its reciprocal basis," *Combinatorica*, vol. 13, no. 3, pp. 363–376, 1993.
- [6] W. H. Mow, "Maximum likelihood sequence estimation from the lattice viewpoint," *IEEE Trans. Inf. Theory*, vol. 40, no. 5, pp. 1594–1600, 1994.

- [7] D. Wübben, D. Seethaler, J. Jaldén, and G. Matz, "Lattice reduction: A survey with applications in wireless communications," *IEEE Signal Process. Mag.*, vol. 28, no. 3, pp. 79–91, 2011.
- [8] F. Eisenbrand, *Integer Programming and Algorithmic Geometry of Numbers*. In M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, and L. A. Wolsey, editors, *50 Years of Integer Programming 1958–2008: From the Early Years to the State-of-the-Art*, pp. 505–559. Springer, Heidelberg, 2010.
- [9] A. Hassibi and S. Boyd, "Integer parameter estimation in linear models with applications to GPS," *IEEE Trans. Signal Process.*, vol. 46, no. 11, pp. 2938–2952, 1998.
- [10] D. Micciancio and O. Regev, *Lattice-Based Cryptography*. Bernstein, D. J. and Buchmann, J. (eds.), Berlin: Springer Verlag, 2008.
- [11] G. Hanrot and D. Stehlé, "Improved analysis of kannan's shortest lattice vector algorithm," in *Proceedings of the 27th annual international cryptology conference on Advances in cryptography*. Springer-Verlag, 2007, pp. 170–186.
- [12] G. Hanrot, X. Xavier Pujol, and D. Stehlé, "Algorithms for the shortest and closest lattice vector problems," in *IWCC'11 Proceedings of the Third international conference on Coding and cryptography*. Springer-Verlag Berlin, Heidelberg, 2011, pp. 159–190.
- [13] O. Goldreich, D. Ron, and M. Sudan, "Chinese remaindering with errors," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1330–1338, 2000.
- [14] V. Guruswami, A. Sahai, and M. Sudan, "Soft-decision decoding of chinese remainder codes," in *Proceedings. 41st Annual Symposium on Foundations of Computer Science, Redondo Beach, CA, Nov. 2000, 2000*, pp. 159–168.
- [15] C. Schnorr and M. Euchner, "Lattice basis reduction: improved practical algorithms and solving subset sum problems," *Math Program*, vol. 66, pp. 181–191, 1994.
- [16] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Math. Comput.*, vol. 44, no. 170, pp. 463–471, 1985.
- [17] X.-W. Chang, J. Wen, and X. Xie, "Effects of the LLL reduction on the success probability of the babai point and on the complexity of sphere decoding," *IEEE Trans. Inf. Theory*, vol. 59, no. 8, pp. 4915–4926, 2013.
- [18] M. F. Anjos, X.-W. Chang, and W.-Y. Ku, "Lattice preconditioning for the real relaxation branch-and-bound approach for integer least squares problems," *Journal of Global Optimization*, vol. 59, no. 2-3, pp. 227–242, 2014.
- [19] J. Wen, C. Tong, and S. Bai, "Effects of some lattice reductions on the success probability of the zero-forcing decoder," *IEEE Commun. Lett.*, vol. 20, no. 10, p. 2031, 2016.
- [20] J. Wen and X. W. Chang, "Success probability of the babai estimators for box-constrained integer linear models," *IEEE Trans. Inf. Theory*, vol. 63, no. 1, pp. 631–648, Jan 2017.
- [21] A. Sakzad, J. Harshan, and E. Viterbo, "Integer-forcing MIMO linear receivers based on lattice reduction," *IEEE Trans. Wireless Commun.*, vol. 12, no. 10, pp. 4905–4915, 2013.
- [22] O. Ordentlich, U. Erez, and B. Nazer, "Successive integer-forcing and its sum-rate optimality," in *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2013, pp. 282–292.
- [23] J. C. Lagarias, H. Lenstra, and C. P. Schnorr, "Korkin-zolotarev bases and successive minima of a lattice and its reciprocal lattice," *Combinatorica*, vol. 10, no. 4, pp. 333–348, 1990.
- [24] B. Helfrich, "Algorithms to construct minkowski reduced and hermite reduced lattice bases," *Theor. Comput. Sci.*, vol. 41, no. 8, pp. 125–139, 1985.
- [25] R. Kannan, "Minkowski's convex body theorem and integer programming," *Mathematics of operations research*, vol. 12, no. 3, pp. 415–440, 1987.
- [26] C. P. Schnorr, "A hierarchy of polynomial time lattice basis reduction algorithms," *Theoret. Comput. Sci.*, vol. 53, pp. 201–224, 1987.
- [27] W. Zhang, S. Qiao, and Y. Wei, "HKZ and Minkowski reduction algorithms for lattice-reduction-aided MIMO detection," *IEEE Trans. Signal Process.*, vol. 60, no. 11, pp. 5963–5976, 2012.
- [28] J. Wen and X.-W. Chang, "A modified KZ reduction algorithm," in *2015 IEEE International Symposium on Information Theory (ISIT)*, 2015, pp. 451–455.
- [29] G. Golub and C. Van Loan, "Matrix computations, 4th," *Johns Hopkins*, 2013.
- [30] X.-W. Chang and Q. Han, "Solving box-constrained integer least squares problems," *IEEE Trans. Wireless Commun.*, vol. 7, no. 1, pp. 277–287, 2008.

- [31] J. Wen, B. Zhou, W. H. Mow, and X.-W. Chang, "An efficient algorithm for optimally solving a shortest vector problem in compute-and-forward design," *IEEE Trans. Wireless Commun.*, vol. 15, no. 10, pp. 6541–6555, 2016.
- [32] J. Wen, L. Li, X. Tang, M. W. H., and C. Tellambura, "An efficient optimal algorithm for integer-forcing linear MIMO receivers design," *arXiv:1610.06618*, 2016.
- [33] L. Ding, K. Kansanen, Y. Wang, and J. Zhang, "Exact SMP algorithms for integer-forcing linear MIMO receivers," *IEEE Trans. Wireless Commun.*, vol. 14, no. 12, pp. 6955–6966, 2015.
- [34] C. Ling, "On the proximity factors of lattice reduction-aided decoding," *IEEE Trans. Signal Process.*, vol. 59, no. 6, pp. 2795–2808, 2011.
- [35] X.-W. Chang, D. Stehlé, and G. Villard, "Perturbation analysis of the QR factor  $r$  in the context of LLL lattice basis reduction," *Mathematics of Computation*, vol. 81, no. 279, pp. 1487–1511, 2012.
- [36] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.
- [37] M. Newman, *Integral Matrices*. Academic Press, New York and London, 1972.
- [38] D. E. Knuth, *The Art of Computer Programming, Vol. 2, Seminumerical Algorithms*. 2nd ed. Reading, MA: Addison-Wesley, 1981, vol. 2, 1981.
- [39] G. H., "On the number of divisions in finding a G.C.D," *Amer. Math. Month*, vol. 31, pp. 443–443, 1924.